World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:7, No:8, 2013

Collaboration of Multi-Agent and Hyper-Heuristics Systems for Production Scheduling Problem

C. E. Nugraheni and L. Abednego

***Abstract*—**This paper introduces a framework based on the collaboration of multi agent and hyper-heuristics to find a solution of the real single machine production problem. There are many techniques used to solve this problem. Each of it has its own advantages and disadvantages. By the collaboration of multi agent system and hyper-heuristics, we can get more optimal solution. The hyper-heuristics approach operates on a search space of heuristics rather than directly on a search space of solutions. The proposed framework consists of some agents, i.e. problem agent, trainer agent, algorithm agent (GPHH, GAHH, and SAHH), optimizer agent, and solver agent. Some low level heuristics used in this paper are MRT, SPT, LPT, EDD, LDD, and MON.

***Keywords*—**Hyper-heuristics, multi-agent systems, scheduling problem.

I. Introduction

SCHEDULING problems, such as production scheduling problems, in real life belong to the complex problems (NP-complete) due to the dynamic and the difficulty for searching the solution. Deterministic searching methods do not work effectively when the problem size is getting bigger. Until now, much research has been conducted to solve the scheduling problem. Several approaches and methods that have been used are heuristics, Simulated Annealing, Hill Climbing, Tabu Search, Evolutionary Algorithms (Genetic Algorithm), Swarm Optimization Algorithm, Artificial Immune System, Variable Neighborhood Search, Hyper-heuristics, Case-Based Reasoning , Fuzzy Reasoning, Agent-Based Methods, Adaptive Learning, and Multi Objective Decision Making.

Most techniques are domain-specific, which means that their applications are fit rather too specific than to general problems. Solving techniques have been selected and adapted manually by humans to solve certain problems. A model is only an approximation of the real problem at a certain time. The performance of the algorithm can be drastically reduced if there is a change in the problem being modeled. Unfortunately, real problems change dynamically and rapidly by nature. This lead to the need for a technique that is easily adapted to a variety of changes.

Hyper-heuristic algorithm provides searching framework that more general and non domain-specific. Hyper-heuristic methodology is more flexible in the search process and can be easily applied to a larger scope of issues [1]. The term hyper-heuristics refers to the heuristics to choose heuristics [2]. This construction of this method is motivated by the need for flexible search techniques, can be easily adapted to respond to changes and free of domain-specific problems. This technique does not directly conduct a search on the solution space, but prior to the heuristic space.

The concept of multi agent systems is an emerging approach in the development of software systems nowadays. A multi agent system is understood as a system consisting of agents (in this case software) that can work independently and communicate and work together to achieve certain goals. Development of systems with this approach has advantages in terms of scalability, extensibility, and distributability.

In this work, a multi-agent framework hyper-heuristics to solve the problem of single production machine scheduling will be developed. With this technique, the concept of multi-agent is adopted to act as a high-level heuristic which is responsible for managing the collaboration of low-level heuristcs.

The remainder of this paper is organized as follows. Section II gives the formal definition of the multi-objective single machine scheduling problem and technique that is often used in solving real scheduling problem. Section III reviews some related works for solving production scheduling problem. Section IV explains the architecture of the proposed framework, in particular the algorithm agents: GAHH agent, SAHH agent, and GPHH agent. Section V gives some concluding remarks and recommendations for future work.

II. Problem Definition

A. Single Machine Scheduling Problem

Single machine scheduling problem is the process of assigning a group of tasks to a single machine or resource [3]. The tasks are arranged so that one or many performance measures may be optimized.

Let CTi, DDi, RDi be the completion time, due date, and the release date of task i respectively, the objective of this problem is to find a schedule that simultaneously satisfies:

1. Minimization of mean tardiness:

$$F_1 = \frac{\sum_{i=1}^{n} \max\{CT_i - DD_i, 0\}}{n}$$

2. Minimization of mean flow time:

$$F_2 = \frac{\sum_{i=1}^{n} (CT_i - RD_i)}{n}$$

Nugraheni, C. E. and Abednego, L. are with Parahyangan Catholic University, Bandung, Indonesia (e-mail: cheni@unpar.ac.id, luciana@unpar.ac.id).

where n is the total number of tasks to be scheduled.

The objective function is constructed by combining the two different objectives into a weighted sum where all the objectives have the same priority. It can be defined as:

$$F = 0.5 * F_1 + 0.5 * F_2$$

B. Heuristic

Heuristic methods are often used to deal with most real-world combinatorial problems which are difficult to solve. These methods have no guarantee of optimality but can produce a solution in a reasonable time even when deterministic method cannot produce one [4].

C. Dispatching Rules

Dispatching rules are among the most frequently applied heuristics in production scheduling, due to their ease of implementation and low time complexity. Whenever a machine is available, a dispatching rule inspects the waiting jobs and selects the job with the highest priority to be processed next [5]. For example, the Shortest Processing Time rule chooses the next job with the shortest time in the queue that will be removed for processing.

D. Hyper-heuristics

Often, heuristics are the result of years of work by a number of experts. An interesting question is how we can automate the design of heuristics. Hyper-heuristics are search methodologies for choosing or generating (combining, adapting) heuristics (or components of heuristics), in order to solve a range of optimization problems [4].

The main feature of hyper-heuristics is that they search a space of heuristics rather than a space of solutions directly. The motivation behind hyper-heuristics is to raise the level of generality at which search methodologies operate. Fig. 1 shows the general framework for hyper-heuristics approach.

III. Related Works

Over the years, there have been several approaches used to deal with various objectives in production scheduling problem. They are:

1. Exact algorithm

Bolat et al. [6] solved machine scheduling problem using Branch and Bound (B&B) technique. Instance data up to 15number of jobs can be solved in a reasonable time. The reported results show that near optimal solution can be found, albeit at the expense of huge computational cost, particularly when the problem size is large.

2. Deterministic heuristics

In practice, dispatching rules have been applied to avoid the computational cost produced by the exact algorithms [7], [8]. Although the quality of solutions produced by dispatching rules is no better than the exact method, they are the more frequently applied technique due to their ease of implementation and their low time complexities.

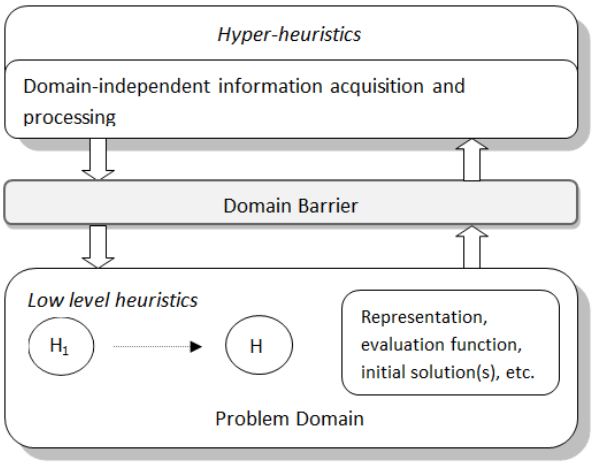


Fig. 1 General framework for hyper-heuristics approach

3. Metaheuristics

The combinatorial nature of most scheduling problems allows the use of search based and enumerative techniques such as genetic algorithms [9]. These methods usually offer good quality solutions, but at the cost of a large amount of computational time needed to produce a solution. Search based techniques are not applicable in dynamic or uncertain conditions where there is need for frequent schedule modifications or changing system requirements.

4. Hyper-heuristics

V´aquez-Rodriguez et al. [10] consider combinations of different dispatching rules to solve a multi-machine cardboard box shop scheduling problems. A standard genetic algorithm was employed as the high level search of sequences of 13 dispatching rules: minimum release time (MRT), shortest processing time (SPT), longest processing time (LPT), less work remaining (LWR), more work remaining (MWR), earliest due date (EDD), latest due date (LDD), weighed shortest processing time (WSPT), weighted longest processing time (WLPT), lowest weighted work remaining (LWWR), highest weighted work remaining (HWWR), lowest weighted due date and highest weighted due date. The hyper-heuristic was shown to be capable of learning effective hybridizations upon dispatching rules during scheduling, and thus was superior to employing single rules for the whole scheduling process.

Abednego [11] investigates the potential use of genetic programming hyper-heuristics for solution of the real single machine production problem. Experimental results show that this technique performs at least as good as the ones produced by man-made dispatching rules. This can be achieved by combine each strength from some different heuristics using members of a set of known and reasonably understood heuristic's components (terminal set and function set).

IV. System Architecture

The proposed system architecture to solve single machine scheduling problem adopt the concept of multi-agent system and hyper-heuristics approach.

A. Agent

There are some agent types in the system: a Problem Agent, a Trainer Agent, Training Dataset Agent, a Heuristics Pool Agent, three Algorithm Agents (GPHH, GAHH, and SAHH), an Advisor Agent, and a Solver Agent. Fig. 2 shows the proposed system configuration. Arrows represent communications between agents.

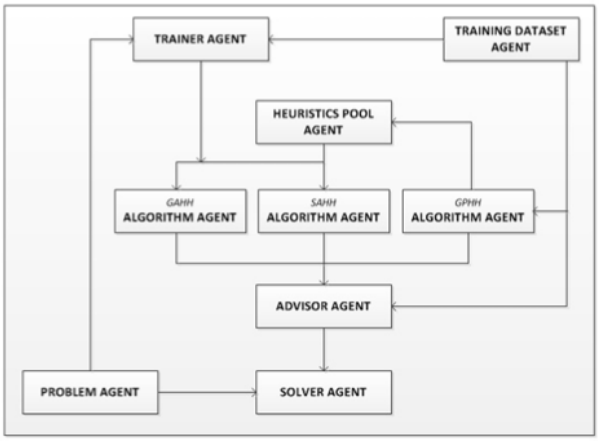


Fig. 2 System's architecture

Problem Agent. This agent is the entry point of the system. The agent initializes all other agents by sending the problem description to the trainer agent.

Trainer Agent. Based on the problem description get from the problem agent, this agent trains the system with a group of training dataset.

Training Dataset Agent. The agent manages the training data set and provides training data set to all algorithm agents through the Trainer Agent.

Heuristics Pool Agent. The agent manages the collection of heuristics (low level heuristics and heuristics produced by GPHH).

Algorithm Agent. The agent is responsible for:

- Running the hyper-heuristics algorithm with received parameter and heuristics
- Sending the best solution found to the optimizer agent after the hyper-heuristics algorithm is finished

There are three Algorithm Agent proposed in this research: GAHH, SAHH, and GPHH. The detail algorithm for each agent can be found in section IV.B-IV.D.

Advisor Agent. The agent is responsible for choosing the best heuristics get from Algorithm Agent (GAHH, SAHH, GPHH).

Solver Agent. The agent solves the problem from the Problem Agent with the best heuristic got from the Advisor Agent. The algorithm for the Solver Agent is given in Algorithm 1.

ALGORITHM 1
SOLVER AGENT'S ALGORITHM

```
while there are unscheduled jobs do
  calculate priorities of all available jobs
  schedule job with the greatest priority first
end while
```

B. Algorithm Agent: Genetic Algorithm Hyper-heuristics

Like other hyper-heuristics approach, Genetic Algorithm Hyper-heuristics works in search space of heuristics rather than a space of solutions directly. Fig. 3 shows a general framework for the Genetic Algorithm Hyper-heuristics used in this research.

First the algorithm creates a random initial population. On each iteration, the algorithm creates population of nindividual. Each individual consists of a range of heuristics selected from the set of low-level heuristics available. The populations are then modified with genetic operation that is chosen probabilistically. When the stopping conditions are met, the system terminates and outputs the best solution found so far. The GAHH algorithm is given in Algorithm 2.

ALGORITHM 2
GAHH ALGORITHM

```
Create the initial random population P of size n
Do
  Evaluate fitness of each individual in the population

  Select genetic operation (reproduction/crossover/mutation) probabilistically

Loop until stopping criteria are met
```

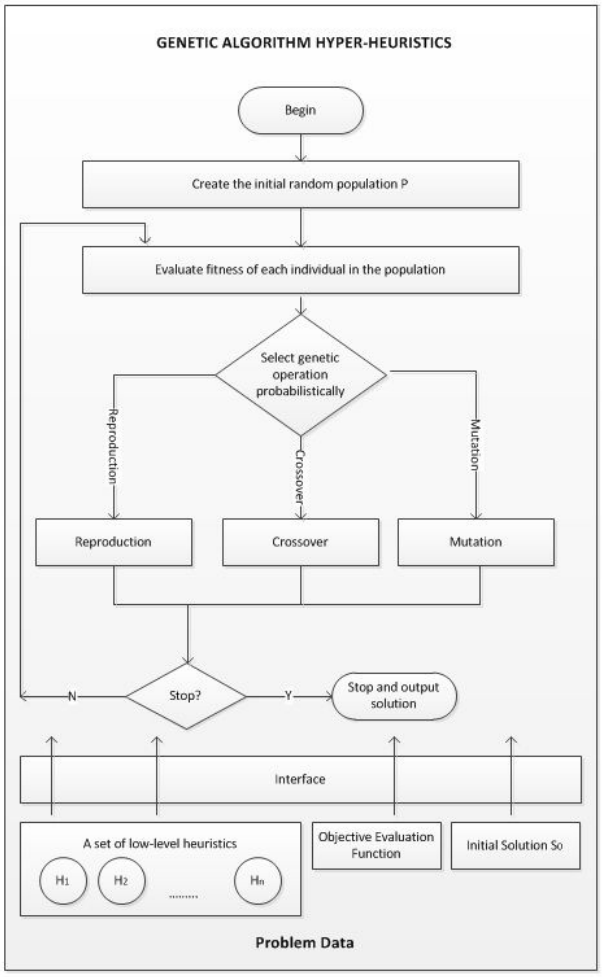


Fig. 3 General framework for GAHH

C. Algorithm Agent: Simulated Annealing Hyper-Heuristics

Simulated Annealing Hyper-heuristics combined Simulated Annealing and Hyper-heuristics approach. Fig. 4 shows a general framework for the Simulated Annealing Hyper-heuristics used in this research. It is the same with the framework proposed by Ruibin Bai et al. [12]

On each iteration, the algorithm selects a heuristic from the set of low-level heuristics available. A heuristic is chosen based on the probability that is associated with a weight. The weight reflects the importance of the corresponding heuristic at the current stage. The weight is dynamically changed based on the performance of its corresponding heuristic. The mechanism to change the weight of heuristic is a penalty-reward strategy. The weight of a heuristic is increased if it produces a better solution and decreased otherwise. A minor positive score is given for those heuristics that cannot improve the evaluation function but still useful in creating intermediate situation to the optimal solution. And a penalty for those heuristics which could neither improves the current solution or generates a new solution. The temperature of the simulated annealing is then modified. When the stopping conditions are met, the system terminates and outputs the best solution found so far. The SAHH algorithm is given in Algorithm 3.

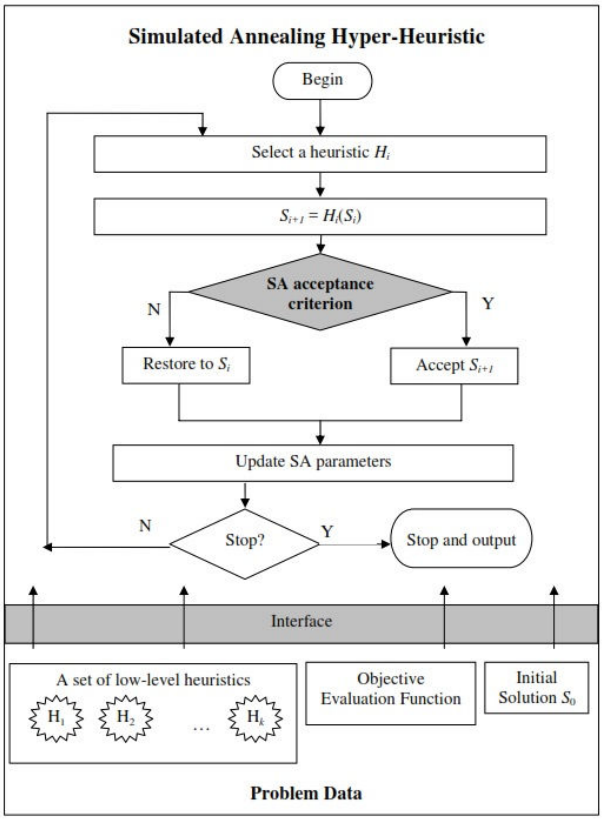


Fig. 4General framework for SAHH

D. Algorithm Agent: Genetic Programming Hyper-Heuristics

Genetic Programming Hyper-heuristics belongs to the family of evolutionary computation methods. Given a set of functions and terminals and an initial population of randomly generated syntax trees (representing programs), these programs are then evolved through genetic recombination (crossover, mutation) and natural selection. A new generation is created by probabilistically selecting individuals from the old generation based on their fitness value. These individuals are either survived intact or genetically modified through a number of operators [2].

Genetic Programming Hyper-heuristics is a form of automatic programming with variable length. The solution is represented by a computer program that takes a number of inputs, i.e. terminal set that are relevant to the problem considered, manipulates them through a number of functions and produces the required outputs. Solution is usually represented in a form of parse tree. Fig.5 illustrates the solution of genetic programming in a form of parse tree. From this parse tree, GPHH-generated dispatching rule is RD + (DD SP).

In GPHH, an individual is composed of terminals and functions. The terminal set and function set that are used in this research are described in Tables I and II. Table III shows some best GPHH-generated heuristics.

ALGORITHM 3
SAHH ALGORITHM

Set initial temperature t_s, stopping temperature t_f, and total iterations k
Generate an initial solution S_0, $t=t_s$
Define a set of heuristic $H_{i(i=0, \ldots, n)}$, assign appropriate weight w_i to each heuristic H_i

Do
Select a heuristic (H_i) based on probability $p_i = \frac{w_i}{\sum_{i=1}^{n} w_i}$
Generate a candidate solution using heuristic H_i
Let δ_i stand for the difference in the evaluation function between s and s′

if$\delta_i > 0$
$s = s'$
$w_i = w_i + k$

else if $\delta_i = 0$ and a new solution is created
$s = s'$
$w_i = w_i + \varepsilon$

else if $\delta_i = 0$ and no new solution is created
$w_i = w_i - \varepsilon$

else if $\delta_i < 0$ and $\exp(\delta_i/t) <$ random(0,1)
$w_i = w_i - k$

if$w_i > w_{max}$
$w_i = w_{max}$

if$w_i < w_{min}$
$w_i = w_{min}$

Loop until stopping criteria are met

TABLE I
TERMINAL SET

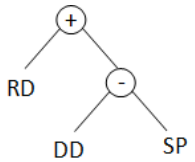
Terminal	Meaning
RD	Release date of a job
DD	Due date of a job
PT	Processing time of a job
W	Weight of a job
N	Total number of job
SP	Sum of PT of all job

TABLE II
FUNCTION SET

Function	Meaning
ADD, SUB, MUL	Addition, substraction, multiplication
DIV	Protected division (DIV(a,b)=1, if |b|<0.000001)

TABLE III
GPHH-GENERATED HEURISTICS

Machine	GPHH Heuristics
GDC	$\frac{DD}{W + ((PT + W) - (W * SP))} - (2 * W + SP)$
PC	$RD * \left\{\left(W + \frac{2 * DD}{N + SP}\right) - (N + PT + DD)\right\}$
Slice	$\left\{\frac{W - PT}{W(3 * DD + PT)}\right\} - RD$
UK60	$W - RD - \frac{N}{DD - (SP^2 * DD)}$
UK75	$\left\{\frac{DD}{SP^2 + PT - W} - N\right\} * RD$



GPHH-generated dispatching rule: RD + (DD-SP)

Fig. 5 An example of a GP parse tree and its interpretation

V. CONCLUSIONS AND FUTURE WORK

We have proposed a framework for solving single machine scheduling problem. The framework combined the concepts of multi-agent systems and hyperheuristics. Three hyper-heuristic techniques used in this work are genetic algorithm, simulated annealing and genetic programming. The architecture of the proposed framework has an advantage that the heuristics generated by GPHH agent can be used by other algorithm agents, GAHH agent and SAHH agent.

It is planned to implement and to apply this framework on a real case study which is the scheduling of single machine problems at a metal industry.

REFERENCES

[1] Burke E. K., Hyde M., Kendall G., Ochoa G., Ozcan E., and Qu R. "Hyperheuristics: A Survey of the State of the Art". 2010.
[2] Burke E. K., Hart E., Kendall G., Newall J., Ross P., and S. Schulenburg. "Hyperheuristics: An emerging direction in modern search technology." In F. Glover and G. Kochenberger (eds.), Handbook of Metaheuristics. Kluwer, pp. 457-474. 2003.
[3] Silva J.D.L., Burke E.K., Petrovic S. "An Introduction to Multiobjective Metaheuristics for Scheduling and Timetabling." 2005.
[4] Burke E.K., Hyde M., Kendall G., Ochoa G., Ozcan E., and Woodward J. "Exploring hyper-heuristic methodologies with genetic programming." In Mumford C, Jain L (eds) Computational Intelligence: Collaboration, Fusion and Emergence, Intelligent Systems Reference Library, Springer, pp 177-201. 2009.
[5] Burke E. K., Hyde M., Kendall G., Ochoa G., Ozcan E., and Qu R. "Hyperheuristics: A Survey of the State of the Art." 2010.
[6] Bolat, A., Al-Harkan, I., and Al-Harbi, B., (2005), "Flow-shop Scheduling for Three Serial Stations with the Last Two Duplicate ", Computers and Operations Research. 2005.
[7] [2] Blackstone J. H., Phillips D. T., and Hogg G. L. "A state-of-the-art survey of dispatching rules for manufacturing job shop operations." In International Journal of Production Research, 20(1), 27-45. 1982.

[8] Oliver, H., Chandrasekharan, R. "E?cient dispatching rules for scheduling in a job shop." International Journal of Production Economics, 48(1), 87-105. 1997.
[9] Man K.F., Tang K.S. and Kwong S. "Genetic Algorithms: Concepts and Design." Springer. 1999.
[10] Vazquez-Rodriguez J.A., Petrovic S., Salhi A. "A combined metaheuristic with hyper-heuristics approach to the scheduling of the hybrid ?ow shop with sequence dependent setup times and uniform machines." In Proceedings of the 3rd Multidisciplinary International Scheduling Conference: Theory and Applications. 2007.
[11] Abednego L. "Genetic Programming Hyper-Heuristics For Solving Dynamic Production Scheduling Problem". 2011.*Proc. ICEEI 2011.*
[12] Ruibin Bai, Edmund K. Burke, Graham Kendall, and Barry McCollum. "A Simulated Annealing Hyper-heuristic for University Course Timetabling." PATAP 2006. pp. 345-350. 2006.